4-2014

CIMLoc: A Crowdsourcing Indoor Digital Map Construction System for Localization

Xiuming ZHANG

Yunye JIN

Hwee Xian TAN
Singapore Management University, hxtan@smu.edu.sg

Wee-Seng SOH

CIMLoc: A Crowdsourcing Indoor Digital Map Construction System for Localization

Xiuming Zhang*, Yunye Jin†, Hwee-Xian Tan† and Wee-Seng Soh*
*Department of Electrical & Computer Engineering, National University of Singapore
{zhangxiuming, weeseng}@nus.edu.sg
†Sense & Sense-abilities Program, Institute for Infocomm Research
{yjin, tanhx}@i2r.a-star.edu.sg

***Abstract*—Indoor maps, as crucial prerequisites for many indoor localization and navigation systems, are sometimes inaccessible. The absence of an indoor map database and the high cost of manually constructing an indoor map produce a need for an inexpensive and efficient way to dynamically construct indoor maps. The ubiquity of sensor-equipped mobile devices enables us to crowdsource user trajectories, out of which indoor digital maps can be automatically constructed at low costs. Similar to other crowdsourced data, the collected user trajectories are often noisy and of low fidelity, which poses a challenge to the accurate map construction. To alleviate this problem, we propose *CIMLoc* - a *c*rowdsourcing *i*ndoor *m*ap construction system for *loc*alization. The system is evaluated with real-world trajectories collected from different mobile devices. We quantify the construction errors by computing the localization errors achieved with the constructed map and the real map. Experimental results reveal that *CIMLoc* is able to construct accurate maps that significantly improve localization results. We believe that *CIMLoc* provides an effective solution to the indoor localization problems where the indoor maps are unavailable.**

I. Introduction

Indoor maps have indispensable uses in indoor localization and navigation, e.g. the location services in a shopping mall and the navigation services in a museum. Although many existing works on localization and navigation [1] [2] are based on the assumption of map availability, indoor maps are often unlikely to be available due to privacy issues. Manually constructing an indoor map through site surveys is both expensive and time-consuming. Furthermore, a map so generated is vulnerable to the external environment changes. The existing works on indoor map construction are confined by either impractical assumptions or excessive infrastructure requirements. Therefore, a practical and inexpensive way to dynamically construct an indoor digital map is of great value and necessity.

Crowdsourcing is a low-cost and efficient way to extract useful information from data acquired from crowd participants. The crowdsourcing concept has been successfully applied to *OpenStreetMap* to collaboratively construct outdoor maps. Unlike outdoor map construction that can utilize Global Positioning System (GPS) locations, indoor map construction has to rely on some other position or trajectory providers. A qualified candidate for this may be the commercially available mobile devices equipped with various sensors (including accelerometer, magnetometer, and gyroscope) and communication modules (such as Wi-Fi and 3G modules).

In this paper, we propose *CIMLoc* - a *C*rowdsourcing *I*ndoor digital *M*ap construction system for *Loc*alization that constructs indoor maps based on crowdsourced user trajectories, and performs localization and navigation for the users. *CIMLoc* comprises three subsystems: (i) Dead Reckoning Subsystem (DRS) that obtains dead reckoning (DR) trajectories from the mobile devices of participating pedestrians; (ii) Map Construction Subsystem (MCS) that constructs the indoor map based on a trajectory partitioning and clustering algorithm [3]; and (iii) Localization and Navigation Subsystem (LNS) that allows users to check their current locations and navigate themselves to their desired destinations via the shortest paths.

CIMLoc assumes that the mobile device can obtain *one* location fix *region* at an entrance, which serves as an anchor point to relate the trajectories for map construction. Note that an accurate location fix point is not required; only an approximation region will suffice for *CIMLoc* to correctly construct the map. This location region may be provided by the loss of GPS signals as the user moves indoors, overhearing of broadcast beacons from existing Wi-Fi routers, overhearing of Bluetooth beacons, or identification of QR codes at known locations. Given the ubiquity of these providers and the limited number of entrances in a building, we believe that this is a reasonable assumption.

The four main contributions of our work can be summarized as follows. First, to the best of our knowledge, *CIMLoc* is the first indoor map construction system that handles the real-world scenario, where crowdsourced DR trajectories can be low-quality. Second, we do not make simplifying assumptions on the layout of the indoor environment as some previous works did. Besides the straight and width-fixed corridors, our system is also able to reconstruct curved and width-varying corridors. Third, our constructed map is a digital map (a collection of points) that possesses two major advantages over a traditional map: firstly, it can be directly utilized for digital uses, such as localization, without any human intervention; secondly, its visualization (image form) can be infinitely zoomed without losing any fidelity, similar to Scalable Vector Graphics (SVG). Finally, we show through implementation and evaluation on mobile Android devices that *CIMLoc* can construct indoor maps with small margins of errors.

The rest of the paper is organized as follows. Section II discusses related work in the literature. Section III describes the *CIMLoc* system in detail. Performance evaluation of *CIMLoc* is presented in Section IV. Section V concludes the paper with suggestions for future work.

II. Related Work

A. Dead Reckoning (DR)

DR, as a fundamental module of a localization or navigation system, updates the current location by adding a displacement vector to the previous location. The three major challenges that lie in DR are the intrinsic sensor noise from the inexpensive sensors, the unpredictable magnetic field distortion caused by indoor metallic infrastructure, and the DR error accumulation nature. Independent DR without external infrastructure aids is only accurate for a short term, and the DR error accumulates to an unacceptable extent easily in practice [2]. Consequently, a DR system is usually aided by some external infrastructure to achieve a satisfactory level of accuracy. Wi-Fi fingerprinting [2], identifiable signatures [4], and map constraints [1] have been proposed to ameliorate the error accumulation situation.

B. Clustering and Trajectory Analysis

Clustering is an effective way to mine out the common features hidden behind a huge data set. Density-based spatial clustering of applications with noise (DBSCAN) [5] is a density-based clustering algorithm that is able to identify and exclude noise during clustering. Its immunity to noise is of great value, especially when noisy data, such as crowdsourced data, are to be dealt with.

Based on DBSCAN, [3] proposed a partition-and-group framework that first partitions trajectories into line segments with Minimum Description Length (MDL) principle, and then performs DBSCAN to those line segments. [3] is able to discover common subtrajectories, which improves the utilization ratio of crowdsourced data.

C. Crowdsourcing Indoor Map Construction

[6] hypothesizes that the walking always starts and ends at the same location. [7] requires users to carry an additional Arduino proximity sensor, and not to take another step until the servomotor finishes one full sweep. [8] assumes that users walk in a straight line in a corridor and that the corridors are perpendicular to each other, and therefore will not suffice for generic indoor environments that may have curved corridors.

Walkie-Markie [9] applies the spring relaxation concept treating the encountered Wi-Fi landmarks as nodes and the user trajectories as springs. As more user trajectories are collected, the node positions are adjusted accordingly so as to achieve the minimum system potential energy. *CrowdInside* [4] uses identifiable acceleration patterns of the elevators, escalators, and stairs to correct DR, and hence constructs the map. This approach will fail in the cases where the users mainly stay on the same floor, e.g. a promotion fair located in an exhibition hall and a one-floor office such as our testbed.

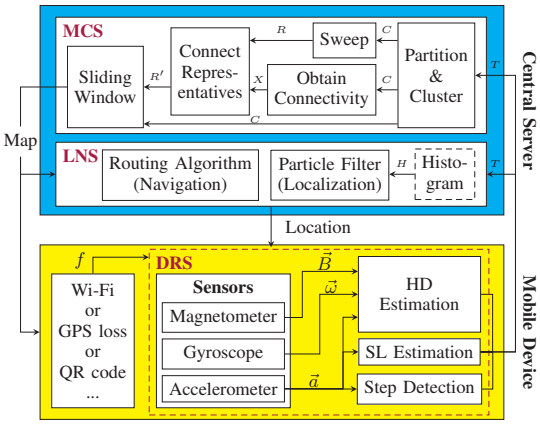


Fig. 1. System Architecture

III. Proposed System

A. System Overview

CIMLoc consists of three subsystems, i.e. Dead Reckoning Subsystem (DRS), Map Construction Subsystem (MCS), and Localization and Navigation Subsystem (LNS). As shown in the system architecture (Figure 1), MCS and LNS are implemented on a central server, whereas DRS is implemented on the mobile devices. The central server crowdsources the estimated trajectories from the mobile devices, constructs the digital map, and notifies the mobile devices of their locations on the constructed map.

DRS The magnetometer, gyroscope, and accelerometer of the mobile device report the magnetic field $\vec{B}$, rotation rate $\vec{\omega}$, and acceleration $\vec{a}$ to the DR algorithm. The step detection algorithm detects the occurrence of each step. The stride length (SL) and heading direction (HD) estimation algorithms characterize each step as a step vector $[l, \theta]^T$, in which l is the SL, and θ is the HD. Given an initial location estimate (x_0, y_0), the series of location coordinates of n such detected steps, $\{(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)\}$, is defined as a trajectory tr in this paper. m such trajectories constitute a trajectory set $T = \{tr_1, tr_2, \ldots, tr_m\}$, which is the input of MCS and LNS.

MCS MCS first partitions the trajectories into segments, and then clusters them into a cluster set C. For each cluster in C, a representative trajectory is generated. The representative trajectory set is denoted by R. Meanwhile, the cluster connectivity is inferred from C, and stored as a connectivity lookup table denoted by X. Given X, representatives in R can be connected, and their intersections can also be found. The connected representatives form the corridor skeleton R', and serve as the input of the subsequent sliding window block to generate the smooth corridor walls W. The output of MCS is the constructed digital map characterized by X, R', and W.

LNS LNS performs particle-filter-based localization and routing-like navigation easily without any human intervention. The particle filter (PF) for localization utilizes the physical constraint that one cannot walk through the walls in estimating the posterior probability distribution of the user's location.

Navigation helps a user find the shortest path from the source to destination.

B. Dead Reckoning Subsystem (DRS)

The algorithm for DRS is shown in Algorithm 1.

1) Step Detection: The acceleration signal is first passed to a low-pass filter. Each step taken by a pedestrian corresponds to a peak-valley (P-V) pair in the filtered acceleration magnitude $\|\vec{a}\|$ pattern [2]. Hence, detecting steps is essentially monitoring the occurrences of valid P-V pairs iteratively (Lines 2-4 in Algorithm 1). A valid P-V pair must have a P-V magnitude difference larger than a threshold value Δ_{PV}, whose value can be determined by calibration. In addition, the occurrence time difference between a valid peak and its neighboring valley has to be greater than another threshold value $\Delta_t = 250\,\text{ms}$. The rationale behind the choice of $250\,\text{ms}$ is that one can only take up to two steps (two peaks and two valleys) per second during a normal-speed walking. Thus, any P-V pair with a time interval smaller than $\frac{1\text{s}}{4} = 250\,\text{ms}$ is unlikely to be valid.

2) Stride Length Estimation: Because each step may have a different SL, we apply the SL estimation model from [10], $l = k \cdot \sqrt[4]{p - v}$, where p is the peak acceleration value, v is the valley acceleration value, and k is a user-specific coefficient. Since Δ_{PV} and k are user-specific, they can both be calibrated by entering the walking distance after 10 steps.

3) Heading Direction Estimation: In this paper, we assume that the users hold the mobile devices with the screen facing up while trying to locate or navigate themselves. With this device placement, HD is the azimuth, and the rotation rate around z-axis ω_z can be easily incorporated to assist the digital compass to improve the HD estimation accuracy. The normal earth magnetic field value is around $42.12\,\mu\text{T}$ [11]. If $\|\vec{B}\|$ exceeds $50\,\mu\text{T}$, $\vec{B}$ is deemed to be distorted, and thus, we use the gyroscope alone to estimate the HD as $\theta = \theta_g = \theta + \int \omega_z(t)dt$, where θ_g denotes the HD estimate from the gyroscope. Otherwise, we use the weighted average of the two HD estimates, given by $\theta = \alpha \cdot \theta_g + (1 - \alpha) \cdot \theta_c$, where α is the weighting constant, and θ_c is the HD estimate from the digital compass.

Algorithm 1 Dead Reckoning Subsystem (DRS)

INPUT: A location fix $f = (x_0, y_0)$; flows of $\vec{a}$, $\vec{\omega}$, $\vec{B}$
OUTPUT: A trajectory $tr = \{(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)\}$
ALGORITHM:

1: **procedure** GENERATETRAJECTORY
2: **while** new $\vec{a}$ **do**
3: $\|\vec{a}\| \leftarrow \text{LPF}\big(\sqrt{a_x^2 + a_y^2 + a_z^2}\big)$
4: **if** detectPV($\|\vec{a}\|$) is TRUE **then** ▷ new step
5: $l \leftarrow k \cdot \sqrt[4]{p - v}$ ▷ SL l
6: $\theta \leftarrow \text{estimateHD}(\vec{a}, \vec{\omega}, \vec{B})$ ▷ HD θ
7: $tr.\text{add}\big((x_{\text{prev}} + l \cdot \sin\theta, y_{\text{prev}} + l \cdot \cos\theta)\big)$

8: **function** ESTIMATEHD($\vec{a}, \vec{\omega}, \vec{B}$)
9: **if** $\|\vec{B}\| < 50\,\mu\text{T}$ **then** ▷ normal value $42.12\,\mu\text{T}$
10: $\alpha \leftarrow 0.99$
11: $\vec{E}_{xyz} \leftarrow \vec{B}_{xyz} \times \vec{G}_{xyz}$
12: $\vec{N}_{xyz} \leftarrow \vec{G}_{xyz} \times \vec{E}_{xyz}$
13: $\theta_c \leftarrow \arctan\left(\frac{E_y}{N_y}\right)$ ▷ from digital compass
14: **else** ▷ distorted
15: $\alpha \leftarrow 1$
16: $\theta_g \leftarrow \theta + \int \omega_z(t)dt$, ▷ from gyroscope
17: $\theta \leftarrow \alpha \cdot \theta_g + (1 - \alpha) \cdot \theta_c$
18: **return** θ

C. Map Construction Subsystem (MCS)

Figure 2 shows an example of the map construction procedure.

1) Partitioning and Clustering: MCS adopts a partition-and-group framework similar to [3], which first partitions the trajectories into line segments using MDL principle, and then groups these line segments into different clusters using DBSCAN [5]. One representative trajectory is generated for each cluster by sweeping a line along the cluster direction.

In [3], the line segment distance for clustering is defined as the simple sum of three basic distance components, namely the perpendicular distance $d_\perp$, parallel distance $d_\parallel$, and angle distance d_θ. In contrast, in this paper, the distance between line segments s_1 and s_2 is computed as a weighted sum

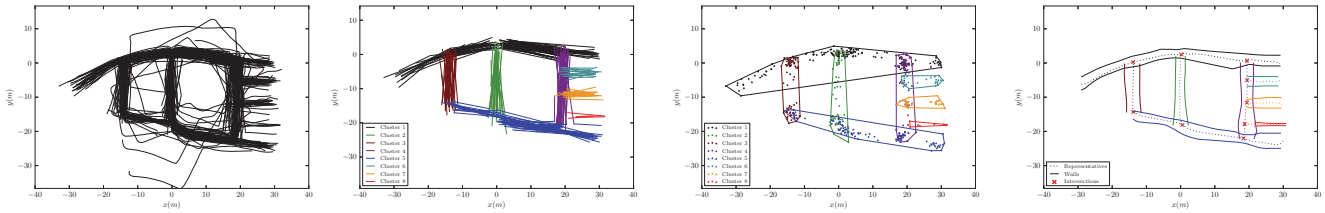
$$d(s_1, s_2) = w_\perp \cdot d_\perp(s_1, s_2) + w_\parallel \cdot d_\parallel(s_1, s_2) + w_\theta \cdot d_\theta(s_1, s_2)$$

where $w_\perp > w_\parallel, w_\theta$. This inequality is imposed because of the following two reasons. First, two line segments may still have a large $d_\parallel$, even though they in fact lie in the same corridor (cluster). Second, there exist parallel corridors that make $d_\theta \approx 0$ in many floor plans. Therefore, under the indoor map context, $d_\perp$ should carry a higher weight than $d_\parallel$ and d_θ so that the clusters become more distinguishable.

DBSCAN is suitable for our problem, because the density concept of DBSCAN effectively excludes noise, the severely drifted trajectories, during the clustering (from Figure 2a to 2b). Current values of the DBSCAN parameters ε and $MinLns$ are respectively set as 1 m and 3 so that the drifted trajectories are excluded, and all the clusters are correctly distinguished.

2) Cluster Connectivity: The partitioning points of a trajectory obtained by MDL principle are essentially its turning points. A turning point means a transition from one cluster (corridor) to another, which implies that these two corridors are connected. Thus, any partitioning point from one cluster falling into another signals the positive connectivity between these clusters. Therefore, the convex hull q enclosing all the partitioning points is found for every cluster (Figure 2c). Then, the connectivity between two clusters c_i and c_j can be easily determined by checking whether any partitioning point of c_i or c_j falls into q_j or q_i, respectively. The cluster connectivity X is stored as a lookup table.

3) Corridor Generation: Incorporating the connectivity information X, we find the intersections and connect the representatives in set R into the connected representative set R', which forms the skeleton of the corridors. The corridor boundaries W, i.e. the walls, are estimated by an averaging window sliding along each $r' \in R'$. The final constructed digital map M containing the corridor connectivity X, corridor



(a) 240 Raw Trajectories (b) Clustered Trajectories (c) Cluster Connectivity (d) Final Constructed Map

Fig. 2. An Example of the Indoor Map Construction Procedure

skeleton R', and corridor walls W is stored in point form as a tiny (12 kB) text file, which can be easily visualized as in Figure 2d.

4) Auxiliary Histogram Algorithm: A crowdsourcing system can perform poorly in its startup phase when the collected data are still sparse. To make the digital map available for localization services as soon as possible, we design a best-effort auxiliary algorithm based on the two-dimensional (2D) histogram for the system startup phase.

We first divide the 2D plane into $1\,\text{m}\times1\,\text{m}$ cells. Once a trajectory point falls into a cell, its cell count increments by one. The cell size $1\,\text{m}\times1\,\text{m}$ is derived from the fact that the average human SL is between $0.6\,\text{m}$ and $0.9\,\text{m}$. On the one hand, a value larger than $1\,\text{m}$ will lead to double counting, if two consecutive short steps remain in the same cell. On the other hand, a value smaller than $1\,\text{m}$ will lead to missed counting, if a long step skips over the cell. Although the histogram method suffers from the drifted trajectories, it provides a best-effort and efficient way of map generation when the trajectories are sparse in the startup phase.

D. Localization and Navigation Subsystem (LNS)

Since X is already available and the edge costs (distances between two adjacent intersections in this case) are also easy to find, a simple routing algorithm, such as Dijkstra's algorithm, will suffice for navigation. Hence, we will omit the discussions on the navigation function, and mainly focus on the localization function.

The algorithm for LNS is shown in Algorithm 2. Note that the starting point (x_0, y_0) as an input is optional for LNS to locate the user. However, in this case where the starting location is not given, the time needed for the particles to converge depends on the trajectory that the user takes.

PF updates the posterior distribution estimates of the current location by observing the physical constraints imposed by the walls. Let p_i^j and w_i^j denote the location coordinates and the weight of the ith particle upon jth step, respectively. PF uses N_p weighted particles $\{p_i^j, w_i^j\}, i = 1, 2, \ldots, N_p$ to approximate the conditional pdf of the location upon jth step as

$$f\left(p^j \mid z^{0:j}\right) \approx \sum_{i=1}^{N_p} w_i^j \delta(p^j - p_i^j)$$

where $\delta(\cdot)$ is the Dirac delta function, and $z^{0:j}$ denotes all the observations until the jth step [2].

Algorithm 2 Localization and Navigation Subsystem (LNS)

IMPORT: Digital Map M; 2D Histogram H
INPUT: n Steps $\{[l_1, \theta_1]^T, \ldots, [l_n, \theta_n]^T\}$; Start (x_0, y_0) (optional)
OUTPUT: Estimated Location $\hat{p}^j$ upon the jth step
ALGORITHM:

1: **procedure** LOCALIZATION
2: **if** (x_0, y_0) is given **then** ▷ Step 1
3: $P^0 \leftarrow \{p_i^0 = (x_0, y_0), w_i^0 = \frac{1}{N_p}\}, i = 1, \ldots, N_p$
4: **else**
5: $P^0 \leftarrow \{p_i^0, w_i^0 = \frac{1}{N_p} \mid p_i^0 \sim \mathcal{U}(M)\}, i = 1, \ldots, N_p$
6: **for** $(j = 1; j \leq n; j$++$)$ **do**
7: **for all** $p_i^{j-1} \in P^{j-1}$ **do**
8: $\theta \leftarrow \theta_j + \theta_v \sim \mathcal{N}(0, \sigma_\theta^2)$
9: $l \leftarrow l_j + l_v \sim \mathcal{N}(0, \sigma_l^2)$
10: $p_i^{j-1} \leftarrow \left(x_i^{j-1} + l \cdot \sin\theta, y_i^{j-1} + l \cdot \cos\theta\right)$ ▷ Step 2
11: **if** p_i^{j-1} in inaccessible region **then** ▷ Step 3
12: $w_i^{j-1} \leftarrow 0$
13: **if** $\exists\ w_i^{j-1} \in P^{j-1}$ such that $w_i^{j-1} \neq 0$ **then** ▷ Step 4
14: $P^j \leftarrow$ resample(P^{j-1})
15: **else**
16: $P^j \leftarrow P^{j-1}$
17: $\hat{p}^j = \sum_{i=1}^{N_p} w_i^j \cdot p_i^j$ ▷ Step 5

In detail, the procedure consists of five steps: initialization, motion updates, measurement updates, resampling, and the expectation computation, among which Step 1 runs only once, and Steps 2-5 run iteratively every time a new step is taken.

First, depending on the availability of the starting position (x_0, y_0), the particles are either all initialized to the starting point or uniformly distributed over the accessible region on the map (Lines 2-5 in Algorithm 2). Second, every particle updates its location coordinates with a step vector $[l, \theta]^T$, which is obtained by introducing zero-mean Gaussian noise into both SL and HD of the new step (Lines 8-10 in Algorithm 2). The proper variance values of the introduced Gaussian noise can be estimated with data obtained from several straight walking trials. Third, all the particles update their weights by examining whether they have violated the wall constraints imposed by either the histogram H in the startup phase or the digital map M in the developed phase. If they do, their weights are set to zero, leading to their eliminations in the resampling step. Otherwise, their weights remain unchanged. Fourth, PF resamples the particles in order to avoid particle

degeneracy. Finally, the location estimate is computed as the expectation of the particles' locations.

Note that in the startup phase when not all the corridors are discovered, it is possible that *all* the particles carry a zero weight. When this happens, they all survive and continue propagating (Lines 15 and 16 of Algorithm 2). Because the particle cloud is spread by a *zero-mean* Gaussian random number, the location estimate $\hat{p}^j$ will roughly follow the DR trajectory. Once any particle gains a non-zero weight by entering the accessible region, Line 13 in Algorithm 2 will be satisfied, causing all the other zero-weight particles to be eliminated immediately. Then, PF restores to its normal operating state, wherein not all particles are of zero weight. With this design, LNS manages to provide services even with sparse trajectories.

IV. Experimental Results

We evaluate the performance of *CIMLoc* using Level 12 of our office building, as illustrated in Figure 3. The mobile

Fig. 3. Testbed - I²R Level 12 South. It contains both straight and curved corridors. The two entrances are labeled in red. The green dots are the waypoints for ground truth collection.

devices used in the experiments are HTC One X and Samsung Galaxy S4.

Our full trajectory set for map construction contains 240 collected trajectories, which all start from either Entrance 1 region or Entrance 2 region, but may take randomly different paths. They neither necessarily follow any waypoints nor traverse one full round.

On the other hand, for localization error analysis, the ground truth is collected as follows. In each trial, a pedestrian starts from Waypoint 1 and traverses the corridors from Waypoint 2 in sequence to Waypoint 10 for three rounds, i.e. $2, 3, \ldots, 10, 2, 3, \ldots, 10, 2, 3, \ldots, 10$. Each trial is approximately 300 m in length. When the user passes a waypoint, the "Stamp" button on the screen is pressed. Thus, each trial contains 27 stamps for localization error computation. 17 of such trials are collected.

Prior to the evaluations, we digitize Figure 3 by manually entering all the necessary corridor wall coordinates into a text file (same data structure as our constructed map). This file serves as the real map to benchmark our constructed map in terms of the localization error, which is defined as the euclidean distance between an estimated location and its corresponding waypoint.

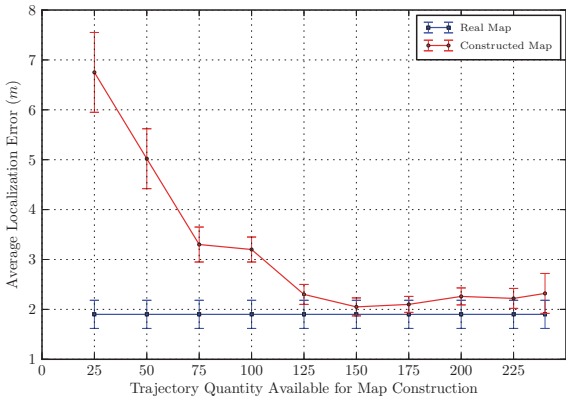


Fig. 4. AMLE vs. the Trajectory Quantity Available for Map Construction

A. Map Construction Performance

We study how the performance of *CIMLoc* evolves as the trajectory quantity increases. Let M_n denote the map generated with n trajectories. MCS iteratively injects 25 random trajectories out of the 240 trajectories to generate a map until all the 240 trajectories are included in the map construction. After this process, a constructed map set composed of $M_{25}, M_{50}, \ldots, M_{225}$, and M_{240} is generated.

We define the average map localization error (AMLE) $\bar{e}_{M_n}$ of map M_n as:

$$\bar{e}_{M_n} = \frac{1}{N_{\text{trials}}} \cdot \sum_{j=1}^{N_{\text{trials}}} \left(\frac{1}{N_{\text{waypoints}}} \cdot \sum_{k=1}^{N_{\text{waypoints}}} e_{M_n}^{j,k} \right)$$

where $e_{M_n}^{j,k}$ denotes the localization error of the jth trial at the kth waypoint when particle-filtered with map M_n. We perform the PF-based localization to the stamped trajectory set with both the real map and the constructed maps. The AMLE together with its 95% confidence interval (CI) is plotted in Figure 4. Note that x-axis values have no effects on the real map curve.

Due to the trajectory sparsity during the startup phase (M_{25} and M_{50}), *CIMLoc* utilizes the auxiliary histogram algorithm to aid the PF-based localization. Figure 4 shows that the AMLE CI starts at a relatively large value, ± 0.8 m, and gradually shrinks to ± 0.2 m. This reflects that the system performance fluctuates in the startup phase due to the trajectory sparsity, but stabilizes soon (M_{75} onwards) as the trajectory quantity increases. As the number of collected trajectories increases beyond 75, the localization performances delivered by the real-map-PF and the constructed-map-PF are less than 0.4 m in difference. This indicates that *CIMLoc* is able to construct a digital map that performs as accurate localization as the real map, implying the high conformity of the constructed map with the real map.

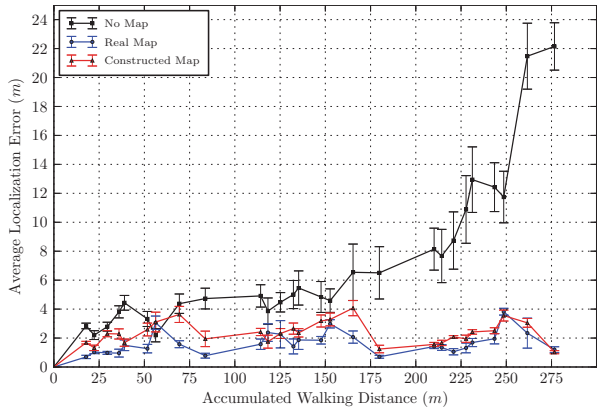


Fig. 5. AWLE without any Map, with the Constructed Map, and with the Real Map vs. the Accumulated Walking Distance

TABLE I
AWLE PERCENTAGE REDUCTION

Distance (m)	85	125	180	227	260
Constructed Map	58.3%	51.1%	81.6%	81.5%	85.3%
Real Map	82.3%	51.8%	86.6%	88.0%	89.9%

B. Map-aided Localization Performance

We evaluate *CIMLoc* in its developed phase, when the system stabilizes and constructs a complete map showing all the corridors, as shown in Figure 2d. We define the average waypoint localization error (AWLE) $\bar{e}_{w_i}$ at waypoint w_i as

$$\bar{e}_{w_i} = \frac{1}{N_{\text{trials}}} \cdot \sum_{j=1}^{N_{\text{trials}}} e^j_{w_i}$$

where $e^j_{w_i}$ denotes the localization error of the jth trial at waypoint w_i. We perform the PF-based localization with the constructed map and the real map. Figure 5 shows the AWLE together with its CI for: (i) no map; (ii) the real map; and (iii) the constructed map.

We observe that both the real map and the constructed map are able to eliminate the DR cumulative errors, and thus significantly reduce localization errors. Figure 5 illustrates that the AWLE by DR alone climbs to 22 m, when the user has traveled for 300 m. Note that a 22 m localization error in an indoor environment has nullified the localization result, because such an error can lead to a totally different corridor (e.g. from Cluster 2 to Cluster 3 in Figure 2b). On the contrary, the PF-based localization with a digital map can successfully mitigate DR error accumulations by constantly correcting the DR trajectory with the map information.

To further quantify the improvements achieved by incorporating the constructed map information, we tabulate the AWLE percentage reductions given by $\frac{\bar{e}_{\text{no map}} - \bar{e}_{\text{map}}}{\bar{e}_{\text{no map}}}$ in Table I. Generally, the AWLE percentage reduction increases with the accumulated walking distance. When the accumulated walking distance reaches 260 m, the percentage reduction is as high as 85.3%. We note that even at the start of the journey, when the error accumulation is not that severe, a discernibly high percentage reduction of 58.3% in AWLE can still be achieved.

To conclude, our proposed system in its developed phase manages to construct an accurate indoor map, with which the DR cumulative errors can be eliminated, and therefore, accurate localization results can be consistently obtained.

V. CONCLUSION AND FUTURE WORK

In this paper, we propose *CIMLoc* - a crowdsourcing indoor digital map construction system for localization and navigation. Experimental results reveal that *CIMLoc* is able to construct the indoor digital map accurately and reduce the localization errors significantly by incorporating the constructed map information into DR. More importantly, our system, as a crowdsourcing system, utilizes the collected data wisely and efficiently, which enables the system to provide the best-effort services when the trajectories are sparse.

As part of future work, we intend to enable *CIMLoc* to tolerate any arbitrary device orientation, as well as to automatically determine the various parameters used in each subsystem. Another possible improvement is to find a deterministic algorithm that triggers the transition from the auxiliary histogram algorithm to the clustering-based algorithm. These improvements will allow the system to be completely independent of human intervention.

REFERENCES

[1] A. Rai, K.K. Chintalapudi, V.N. Padmanabhan, and R. Sen. Zee: zero-effort crowdsourcing for indoor localization. In *Proceedings of the 18th annual international conference on Mobile computing and networking*, Mobicom '12, pages 293–304, New York, NY, USA, 2012. ACM.

[2] Y. Jin, W.S. Soh, M. Motani, and W.C. Wong. A robust indoor pedestrian tracking system with sparse infrastructure support. *Mobile Computing, IEEE Transactions on*, 12(7):1392–1403, 2013.

[3] J. Lee, J. Han, and K. Whang. Trajectory clustering: a partition-and-group framework. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, SIGMOD '07, pages 593–604, New York, NY, USA, 2007. ACM.

[4] M. Alzantot and M. Youssef. Crowdinside: automatic construction of indoor floorplans. In *Proceedings of the 20th International Conference on Advances in Geographic Information Systems*, SIGSPATIAL '12, pages 99–108, New York, NY, USA, 2012. ACM.

[5] M. Ester, H. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. pages 226–231. AAAI Press, 1996.

[6] Y. Xuan, R. Sengupta, and Y. Fallah. Making indoor maps with portable accelerometer and magnetometer. In *Ubiquitous Positioning Indoor Navigation and Location Based Service (UPINLBS), 2010*, pages 1–7, 2010.

[7] D. Vavili, D. Gudlur, P. Vyas, F. Luqman, and P. Zhang. Smilas: sensor based map generation for indoor location aware systems. In *Proceedings of the Fourth ACM SIGSPATIAL International Workshop on Indoor Spatial Awareness*, ISA '12, pages 33–36, New York, NY, USA, 2012. ACM.

[8] H. Shin, Y. Chon, and H. Cha. Unsupervised construction of an indoor floor plan using a smartphone. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 42(6):889–898, 2012.

[9] G. Shen, Z. Chen, P. Zhang, T. Moscibroda, and Y. Zhang. Walkie-markie: indoor pathway mapping made easy. In *Proceedings of the 10th USENIX conference on Networked Systems Design and Implementation*, nsdi'13, pages 85–98, Berkeley, CA, USA, 2013. USENIX Association.

[10] H. Weinberg. *Using the ADXL202 in Pedometer and Personal Navigation Applications*, 2002.

[11] National geophysical data center geomagnetic calculators. http://www.ngdc.noaa.gov/geomag-web. Accessed: 2013-10-24.